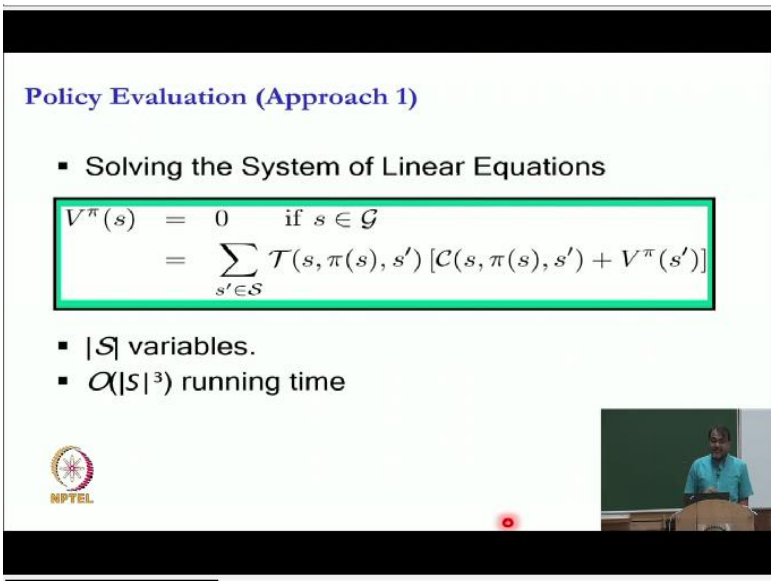
Artificial Intelligence
Prof. Mausam
Department of Computer Science and Engineering
Indian Institute of Science Education and Research-Delhi

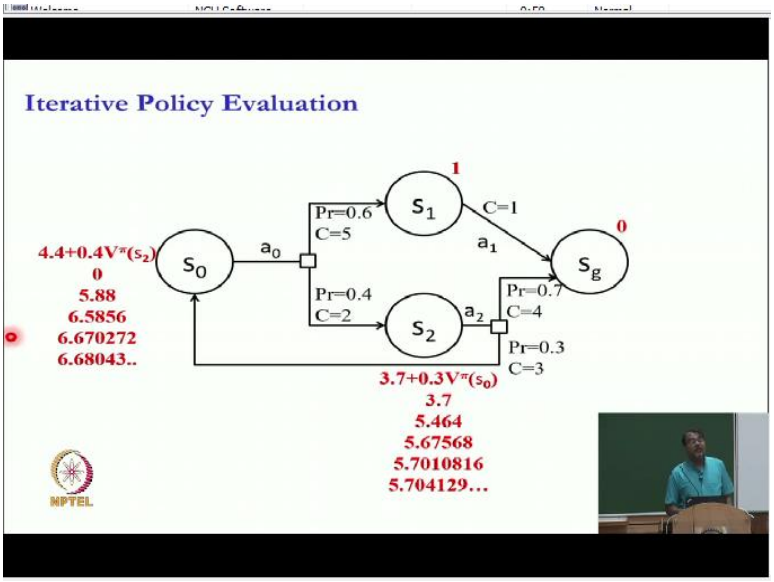
Lecture-73
Markov Decision Processes:
Iterative Policy Evaluation
PART-4

(Refer Slide Time: 00:17)



Now, what I am going to do is I am going to give you a second algorithm to solve the same problem. And the second algorithm is going to be called iterative policy evaluation.

(Refer Slide Time: 00:28)



And the reason I am giving you this algorithm because it will build towards what I want to get to, which is the value iteration algorithm. So same problem, I have also computed value of goal to be 0, value of S 1 to be 1 in this case, but I have a problem in computing value of S 0 and value of S 2. I know that value of S 2 is determined by 3.7 + 0.3 times V pi of S 0. I know the value of S 0 is determined by 4.4 plus-+ 0.4 times V pi of S 2.

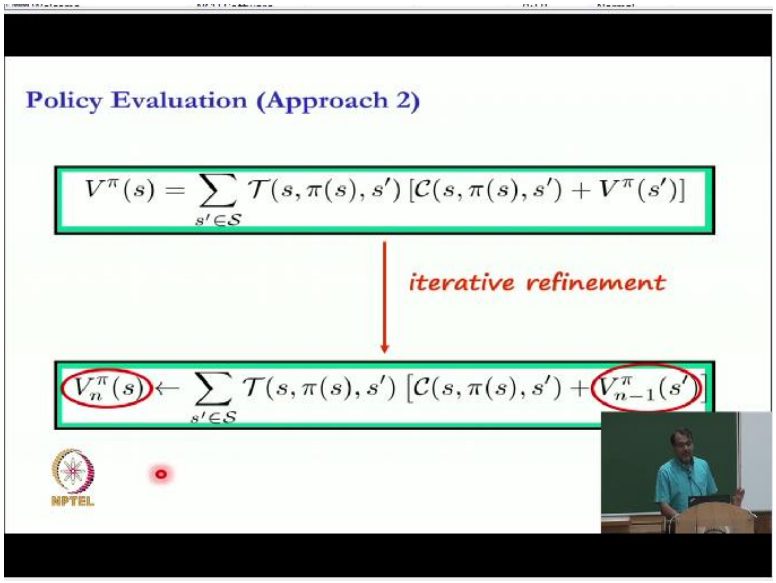
You can check these numbers, right. Now, a simple way to do this is that I randomly initialize these values. So I just say S 0 is the goal, it has value 0. But if I know S 0 then can I compute S 2 using the second equation 3.7 + 0.3 times S. So what is my new value S 2 3.7. Now something happened. S 2 has new value, its value is 3.7. So that changes the value of S 0, what is the new value of S 0 something 4.4 + 0.4 times 3.7, which is 5.88.

Now S 2 says, dude you change your value, now I have to change my value, so I change my value to 5.464. Same thing happens with S 0 it says no, my successor has changed its value, I am going to change mine. So it becomes 6.58. Then S 2 say the same thing, it becomes 5.67. It says the same thing is become 6.67. It changes, it changes, it keeps on going, what do you observe, that at least for this tiny example, they seem to be converging.

And in fact, you can prove that they will converge under some assumptions, which we are not going to talk about. The assumption is that everything should be well formed. Every value

function should be well formed. And what does that mean. We will leave it to an advanced course. This algorithm is going to be called iterative policy evaluation.

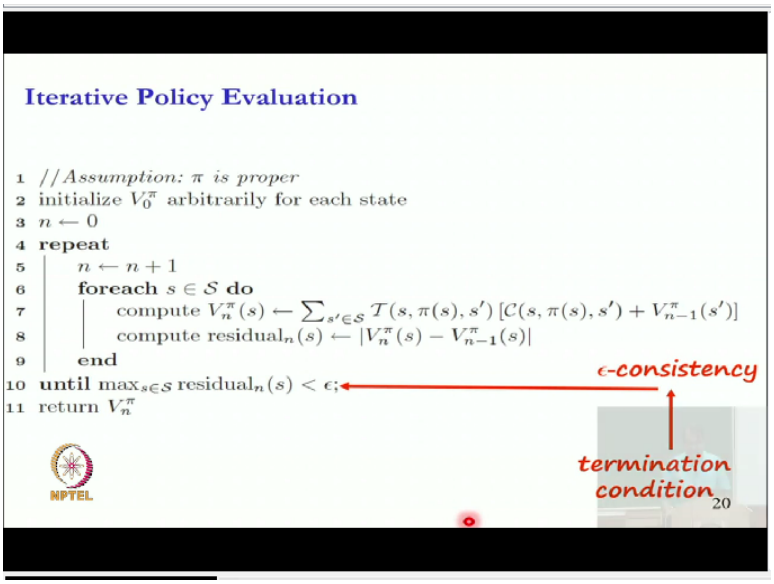
(Refer Slide Time: 03:10)



It basically says that if I want to compute the system of equations, which are given in this box I will randomly initialize them and approximate them by V pi of 0. And, in fact, if I am given an approximation V pi n - 1, which is n – 1th approximation for the V pi function, then in the right hand side, I will use the previous approximation, and I will compute the new approximation V pi n.

And this n sort of denotes the number of approximation. Just to clarify, it is the same equation, but on the right hand side I use the n – 1th approximation, on the left hand side I compute the nth approximation. And this algorithm would be called the iterative refinement version of computing the system of equations, or you can call it iterative policy evaluation. So, this is it.

(Refer Slide Time: 04:18)



So this is your pseudo code. Let us work with this together. So that we are in it we understand this. So there is one assumption which I was trying to hide under the rug. But I will just quickly say the assumption is that your policy pi is proper, proper means that eventually you will reach the goal with every from every state. If that does not happen, think about what happened. So with 0.2 probability, I do not reach the goal starting in state S.

Then what happens it is V pi becomes infinite. It is V pi becomes infinite because it becomes 0.2 times infinity + 0.8 time some finite cost and that we do not like. So we will assume that my policy pi is proper. That means that from every state I will finally reach the goal. I do not know how many steps but I will finally reach the goal, in definite horizon. So, in that assumption this algorithm works, it basically says you can initialize V 0 pi arbitrarily.

Finitely, just do not do infinity minus infinity, you can do it all 0s, all 2's, all 55, all in some 1, some minus 1, some 35, some minus 35, it does not matter. The algorithm is so robust that you can give it any initialization and it will work make sure the goals are 0. Do not mess with the goals. So, value of goal is 0, everything else you can initialize as you want. Then, what you do is you repeat the following steps in every iteration n.

For every state you use so, this is iteration n, for every state you compute the new approximation using the equation that we described earlier in the last slide. So, here I will be using the n – 1th

approximation and I will be computing the nth approximation, now when I do this I maintain a previous set of values and I maintain the new set of values such an algorithm is called the somebody know what kind of an algorithm it is, it is a dynamic programming algorithms very good.

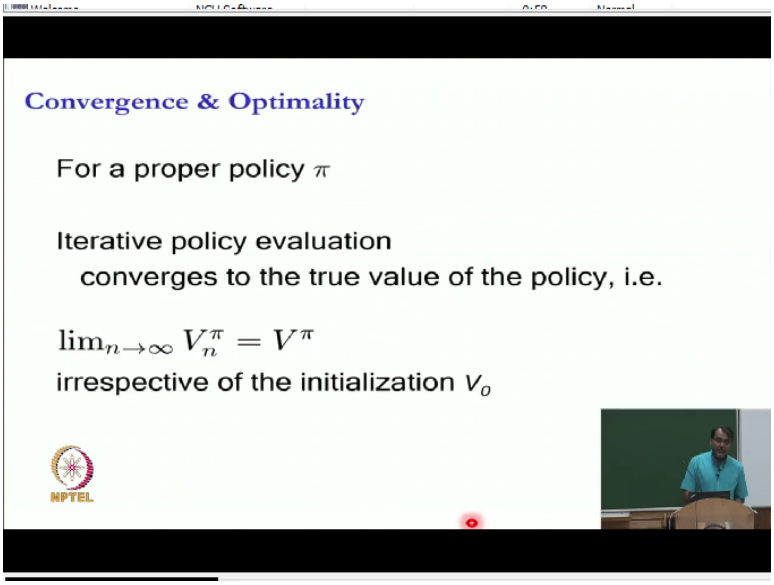
Now you will think about where is the repeated computation and so on. I will let you think about, this as the dynamic programming algorithm. Now, the problem is that I need to stop at some point and my notion of stopping should be that my values have converged. What do mean the value is converged in this case. So what do we want. We want the nth approximation to be very close to the n – 1th approximation.

For every state; not just for me; but for every state in fact, think about it, if it is not true for every state if it is not true even for one state, then it is possible that its predecessor will suddenly change its value in the next iteration significantly. So, we want this to happen for every state. So, we will also compute a phenomenon called residual, residual basically says what is the magnitude of the difference between V pi of n - V pi of n – 1.

Here I am computing residual for every state, but then I will say that you will stop when your max residual is less than epsilon and such a value function will be called the epsilon consistent value function, it is not the optimal value function, it is epsilon close to the optimum and you can actually have a bound on how close you are if you have a epsilon consistency, we will not talk about the bound for now.

But it is close and this is my termination condition and you written the value function V pi n, which was the question at hand, how do we compute the value of a policy.

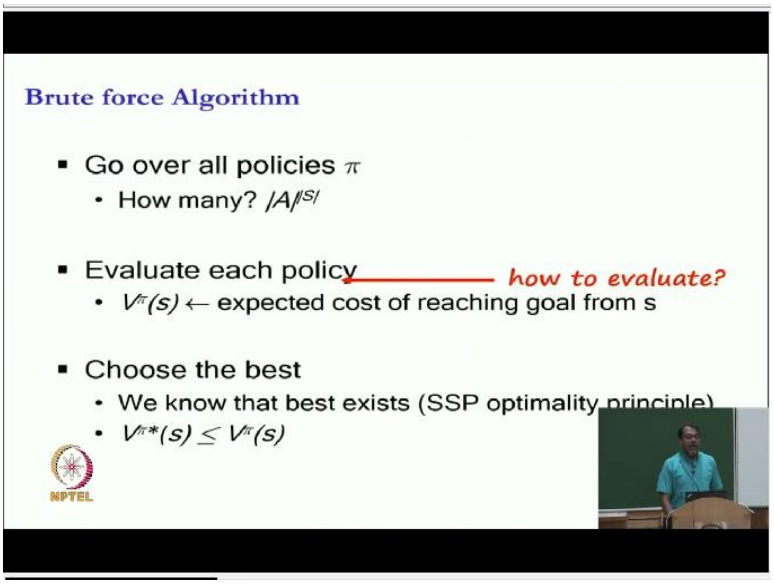
(Refer Slide Time: 08:32)



And moreover, you can prove that for a proper policy P, a pi iterative policy computation converges to the true value of the policy that is limit n tends to infinity, V pi of n is equal to V pi irrespective of the initialization V 0 for every states. This all only happens for the proper policy and this happens in the limit, question. Is there experimental and theoretical evidence on how fast it converges and absolutely.

And you can say that the number of iterations you will take will be polynomial in epsilon 1 over 1 minus gamma and S. So, there is a huge theory on this, right. This model came out in the 50s okay, so that we are really, we are really, really behind in our understanding of the world right. After 50 years lots of things have happened. Now, I want to take the leap. And the leap is we only started with a brute force algorithm, right. The brute force algorithm we started with was that you start with some you enumerate all policies pi.

(Refer Slide Time: 09:56)



And then you evaluate the policy of the pi and then pick the best. Now, we have solved the brute force problem, because now we know how to evaluate a given policy. But that is not sufficient for us, right because we cannot enumerate an exponential number of policies, that is just not going to happen. So, what we will do is we will take this an intuition of iterative policy evaluation and convert it into an algorithm called the value iteration algorithm, which would go beyond the brute force.